Regnecentralen
Smallegade 2, F
2. nov. 1962

GIER ALGOL RESTRICTIONS AND STANDARD PROCEDURES.

RESTRICTIONS IN DASK ALGOL WHICH DO NOT EXIST IN GIER ALGOL.

The following notes refer to A Manual of the DASK ALGOL Language. The following restrictions in DASK ALGOL do not exist in GIER ALGOL:

Sections 7.1.3 and 7.6.1: The integer divide operator is fully available. The hardware symbol is : Likewise the implication operator (7.1.3 and 7.7) is available, the symbol is =>

Section 7.3.1: Identifiers of any length will be fully recognized in GIER ALGOL.

Section 7.9: A subscripted variable is a permissible variable as the controlled variable in a for clause. The identity of the variable will be established once at the beginning of each activation of the for statement and changes of the values of subscript expressions in the course of the execution of the controlled statement will have no influence on which variable is used as the controlled one.

Sections 7.10.1 and 7.13.1: Recursive procedures will be processed fully in GIER ALGOL.

Section 7.11: In GIER ALGOL declarations may appear in any order in the block head.

STANDARD PROCEDURES IN GIER ALGOL.

The new version of GIER ALGOL which is now in preparation will include all the standard procedures included in DASK ALGOL and in addition the following new procedures:

NEW STANDARD OUTPUT PROCEDURES.

STANDARD PROCEDURES: tryktegn, skrvtegn.

Syntax.

<tryktegn statement> ::= tryktegn(<arithmetic expression>) |
skrvtegn(<arithmetic expression>)

Examples.

tryktegn(if upper case then 60 else 58)
skrvtegn(49)
skrvtegn(symbol - case)

Semantics.

The execution of a tryktegn statement causes the character corresponding to the value of the actual parameter to be outputted. The correspondence between the integers and the characters is given in the table of section 6.5. If the value of the actual parameter is not an integer it will be rounded to the nearest integer. Case shifts must be outputted explicitly. NOTE THAT ALL OUTPUT PROCEDURES IN GIER ALGOL ARE EXPECTED TO LEAVE THE EQUIPMENT IN LOWER CASE.

NEW STANDARD INPUT PROCEDURES.

STANDARD PROCEDURES: tast, taststreng.

These procedures are entirely similar to procedures læst and læsstreng (sections 9.5 and 9.6) but expect the input character to be typed on the typewriter.

138a

STANDARD PROCEDURES: læstegn, tasttegn.

Syntax.

<læstegn function designator> ::= læstegn | tasttegn

Examples.

if tasttegn = 49 then go to a
symbol := læstegn

Semantics.

læstegn and tasttegn are integer procedures having an empty formal parameter part. Each call of a læstegn function designator will activate the corresponding input unit (paper tape reader for læstegn, typewriter for tasttegn) and will return with the value of the next proper character from the input medium as its value. By proper character is here meant a character which is not handled by the universal input mechanisms (section 9.2). The values of proper characters in lower case are given directly by the table in section 6.5. In upper case the value supplied by læstegn and tasttegn is increased by 128. Thus the letter p will appear as 39 while P will be 167.

STANDARD PROCEDURE: sættegn.

Syntax.

<sættegn statement> ::= sættegn(<arithmetic expression>)

Examples.

sættegn(160)
sættegn(tegn)

Semantics.

Each call of sættegn assigns the value of the expression supplied as actual parameter to an internal buffer and at the same time sets an internal Boolean variable which causes the value in the buffer to be used as the first proper input character at the first following call of any input procedure (læs, læst, tast, læsstreng, taststreng, læstegn, tasttegn, trykkopi, skrvkopi) ahead of the next symbol waiting in the input unit.

The values of the actual parameters supplied in calls of sættegn should only be such which correspond to proper input characters, i.e. such which may appear as values of læstegn.

STANDARD PROCEDURE: tegn.

Syntax.

<tegn function designator> ::= tegn

Examples.

if tegn < 10 then tryktegn(tegn)
if tegn = 133 then go to exit

Semantics.

tegn is an integer procedure having an empty formal parameter part. Its value is the number corresponding to the last proper character previously inputted by any standard input procedure (læs, læst, tast, læsstreng, taststreng, læstegn, tasttegn, trykkopi, skrvkopi) or assigned by sættegn. The value corresponding to a proper character is to be understood in the same sense as for procedure læstegn. Note that tegn does not activate any input unit, but only makes the last character supplied by any input unit available.

138a

TRANSFERRING VALUES TO DRUM.

STANDARD PROCEDURES: til tromle, fra tromle.
STANDARD VARIABLE: tromleplads.

Syntax.

<drum transfer function designator> ::= til tromle(<array identifier>) |
fra tromle(<array identifier>)
<tromleplads variable identifier> ::= tromleplads

Examples.

```
Bplads := tromleplads
Bshift := til tromle(B)
tromleplads := tromleplads - Bshift
fra tromle(B)
```

Semantics.

The standard integer procedures til tromle and fra tromle and the associated standard integer variable tromleplads administer the handling of transfers of arrays of values to and from the drum memory of GIER, which in this context acts like a special set of input-output organs. The procedure til tromle will transfer the array of subscripted variables identified in the actual parameter to the drum and acts like an assignment of values to the drum and likewise the procedure fra tromle will assign values previously transferred to the drum to the array identified in the actual parameter. In either case the part of the drum involved in the transfer is defined by the value of the integer variable tromleplads which enters into til tromle and fra tromle as a non-local identifier. Thus in order to retrieve a set of values previously transferred to the drum the procedure fra tromle must be called with tromleplads having the same value as when the corresponding call of til tromle was made. The same holds if it is desired to assign new values to a previously used section of the drum. In any case the array supplied as parameter in the drum transfer function designator must be of the same type and have the same number of subscripted variables as the one used in the corresponding call of til tromle. However, the two arrays need not have the same number of subscripts or the same subscript bounds. If the arrays differ in these respects the correspondence of elements is established by ordering the elements of each array in the same manner as they would be if they were read from tape by means of the standard procedure læs (cf. section 9.4.3.2).

Clearly the standard variable tromleplads is the key to administering values stored on the drum. In addition the programmer may use the values of the drum transfer function designators. These are closely related to tromleplads as apparent from the following four rules which define the behaviour of the value of tromleplads:

1. tromleplads is initialized by the compiler to a value which is the one extreme of its permissible range of variation.

2. Every call of til tromle and fra tromle will, as a side-effect, change the value of tromleplads in a direction away from the initial value

supplied by the compiler towards the other extreme of its permissible range and by such an amount that the new value is the correct one to use in transferring values to the next adjecent section of the drum.

3. The amount by which tromleplads is changed through a call of til tromle or fra tromle will be the same whenever arrays of the same type and having the same number of subscripted variables are transferred. The amount by which tromleplads is changed is available as the value of the drum transfer function designator. In other words:

new value of tromleplads = old value + til tromle(A)

new value of tromleplads = old value + fra tromle(A).

However, nothing further about the dependence of the change of tromleplads on the size and type of the array is defined generally (the precise meaning of tromleplads may change from one edition of the compiler to another).

It will be understood from these rules that as long as no explicit assignment is made to tromleplads only calls of til tromle will be in order and each of these will use a new section of the drum adjecent to the one used in the last previous call of til tromle. Before any call of fra tromle is made the programmer must make an explicit assignment to tromleplads. The values assigned to tromleplads can only be derived from its previous values possibly modified by integral multiples of the amount by which is has changed.

The programmer has his full freedom to overwrite sections of the drum which have previously been used as long as he makes sure to use only values of tromleplads which lie within the range defined by its initial value and another extreme which marks the other end of the free section of the drum. If tromleplads steps outside this range an error reaction will occur at run time and the message

tromle ak

will be typed. The criterion for a set of values previously transferred by til tromle to be still intact on the drum may be formulated as follows: Each section used on the drum by til tromle will be defined by an interval of the values of tromleplads, namely that defined by the value of tromleplads just before til tromle was called and its value just after the call was completed. The values transferred will still be intact as long as no call of til tromle with an overlapping interval of tromleplads has been performed.

The version of til tromle and fra tromle now in preparation will in each call use a full number of drum tracks. Consequently arrays of from 1 to 40 variables will require 1 drum track, arrays of from 41 to 80 variables will require 2 drum tracks, etc. The tracks available for values are the ones not used for standard procedures and other permanent programs or for the program itself. In the version of the compiler now being prepared the free tracks will be the ones from about track no. 63 up to the track below the first program track. The program is stored in consecutive trakcs from track 319 and downwards. Values will be transferred first to tracks not used by the compiler so that programs using only a moderate part of the drum will leave the compiler intact. If the compiler has been destroyed by values, entry into it after program slut will be suppressed and instead the message

tromle ak

will be typed.

Peter Naur.